Informal Comments
on the PGS Toolkit Study Report

Thomas E. Goff
NASA/GSFC/MODIS/SDST/RDC
19 October 1992

Reference: EOS Science Software Developer's Handbook, section 3110

The comments in this document contain the requested user feedback from the point of view of the MODIS Level-1A and 1B designs and represent these designs as they have been currently defined. The inclosed comments have been limited to interactions with the MODIS Level-1A and Level-1B data product generators. Section titles roughly follow the numbering convention of the above referenced document.

General comments

Consideration should be given to the passing of function (subroutine) arguments as pointers (addresses) to structures (arrays), rather than actual values. A prototypical function argument list could then contain three starting addresses of: an input structure containing user specified values, an output structure with toolkit supplied values, and a control or information structure containing auxiliary information about the toolkit assumptions. For example, a time request might contain an input structure with the user specified time frame (GMT, sideral, offset, format etc.), the output structure might contain a character string with the requested time in addition to an array of integers with hours-minutes-seconds-etc, and the control structure might contain the revision of the toolkit function and the source and/or accuracy of the time obtained. Structures can be standardized with PGS toolkit supplied, system wide 'include' files or object oriented methods (object classes) and can be utilized in C, FORTRAN, and C++ compilers which will comply with ANSI C, FTN90, and C++ standards.

Data product generators that will be written according to the PGS guidelines can be delivered more quickly if a staged delivery of the PGS toolkit can be performed. Early delivery of toolkit functions to the SCFs will be necessary to allow a more uniform development effort to be accomplished. Toolkit delivery is the major driving milestone for delivery of science and data product generation code.

3.3.1 Production Control and Scheduling Tools

This is the area which can be least specified at the present time and therefore need the most study. A 'wish list' of criteria for the design of the scheduler would be a good addition to the toolkit preliminary specifications. For example: although the PGS is primarily a data driven system, a provision for initiating a chain of processes such that a final product in the data chain is completed within a certain time needs to be added to this criteria list.

Presumably, the metadata associated with a data product (ancillary data in the Level-0 case) will be sufficiently comprehensive that the data set sparcity and robustness can be determined by the requesting data product generation program at the time the generator is initiated. The scheduler must have the ability to initiate programs based on a 'fuzzy' criteria, rather than a go/nogo decision. The product generation program must have the ability to transmit this 'fuzzy' criteria into an implementation, in the scheduler, of a dependencies matrix that will optimize the use of machine resources while minimizing reprocessing due to changes in the indicated comprehensiveness of the metadata. Processing may be performed even if part of an input data set is not available and the data is required for subsequent product generators. Processing may also be performed using alternate ancillary data sets if the primary ancillary data is incomplete or not available in a timely manner.

Communication between the data product generator and the PGS should be in the form of messages that can be accessed by the product generator in an asynchronous manner. Partial execution can be handled by including the expected volume of output product within the data product initiation message. Partial product production implies the need for the product generator to track the quantity and location of the output products within a data set and to have direct access to components of the output product data set.

3.3.1.1 Data Availability Tools

A distinction needs to be made between the input data product and ancillary data required to produce an output data product. Ancillary data availability and quality may not be known upon the initiation of a data product generator, but the input Data Product range is known a-priori by the scheduler as this is primarily a data driven system. Input Data Product metadata is assumed to be available concurrently with each input Data Product. Ancillary metadata has yet to be determined and will be a function of each ~~meta~~data type.

The MODIS data product design assumes that a PGS facility will be provided that will allow the various programs in the processing chain to write to a time sequential log file which may possibly be unique to MODIS. The intent is to provide an accountability mechanism that can be examined by any parties interested in the MODIS (or other) processing. This log file is to be handled by the PGS such that it can be appended to, while being read by, other processes.

3.3.1.2 Initialization Tools

The stream IDs are necessary to provide the forward and backward pointers that specify uniquely, all the inputs and outputs, including ancillary and metadata, for each link in the processing chain.

Computer resources, such as memory and disk space, need to be able to be preallocated by each data product generator before a product can begin to be produced. Performance considerations dictate that these resources may be made physically (directly) available upon program request without any logical to physical translation. (The ability to bypass virtual or backing store memory techniques.)

3.3.1.3 Termination Tools

Note that metadata as part of the input Data Product is read only, and that the DAAC output Data Product metadata is equal to the input metadata with the current Data Product metadata appended.

The ability to synchronize the sharing of computer memory between linked processes in the processing chain may allow Data Products to be produced more efficiently. This implies a semaphore, piping, fork/tee, class I/O, and/or threading type of facility to allow for the passing of resources among processes. The protections that need to be applied to these common data areas are similar to the normal disk protections as applied to maintenance and transfer of ownership, and user selectable protections on at least an owner, group, system level. Good analysis tools to monitor and resolve conflicts within all aspects of these schemes are essential.

3.3.2 I/O Tools

The MODIS design assumes that a process has the ability to request (or otherwise obtain) access to user specified, fixed file record lengths. This facility will increase the access time without necessarily resorting to backing buffers. Note that the MODIS design will handle whole scan cubes of data that will be approximately

2.5 Megabytes in length. The ability to fetch a data buffer directly without intermediate buffering would be desirable from a performance viewpoint. Perhaps these capabilities are what is meant by high speed access.

3.3.2.1 (and subsections) Temporary File I/O Tools

I would have thought that temporary data files would be managed only by the individual programs that need the temporary access. This implies that the format (structure) of those temporary files is best determined and managed by the creating program and not the toolkit. Therefore, I see no need for temporary write_image, read_image, etc. predefined data structures. This also applies to production stream files but does not apply to Data Product files.

What determines whether a file is a "production stream file"?

3.3.2.2 Production Stream File I/O Tools

If these files are to be maintained by the DAACs independently of the local file storage then a tool kit function may be needed. Otherwise, the comments in the previous section apply.

3.3.2.3 Product and Auxiliary Data Access Tools

These file access functions are used to interface the data product generators with the DAAC and other product generators in a processing chain (stream) that produce or use Data Products. File structures that will be common across different Data Products should be defined at the tool kit level. These may include a metadata structure with free form information for comments and rigid structures for temporal and spatial coverage, for example. A common browse structure such as TIFF or GIF formats, if browse is to be generated by the data product generation programs, may also be a tool kit facility. Auxiliary Data of a general nature (DTM, position and attitude, etc.) was discussed in the introduction of this writeup. Care needs to be exercised to provide common interfaces to all available computer languages.

Mention was made in this section that metadata and browse are passed to the IMS. MODIS is expecting the metadata to be associated with each Data Product and would therefore be concurrent with the Data Products. Metadata is only appended to, not changed, by each segment in the processing chain.

3.3.2.3.1 Level 0 Data Access Tools

In a purely data driven system, processes are started when the data are available. The scheduler determines the time period of raw data from the known input data volume and output Data Product timeliness. The data product generator interrogates the PGS to verify the level-0 data availability and completeness, and the existence and validity of any primary and secondary auxiliary data sets. MODIS is planning to use the packet data, not data reconstituted to the instrument output data stream (definition?).

3.3.2.3.2 Level 1 - 4 Product Access Tools

MODIS is expecting to specify the structure of the MODIS Data Products. This will be well documented as part of the CASE design of the MODIS processors and will be available to all interested parties both at the design stage and at the implementation stage as include headers or class structures as to be determined (TBD). It is not expected, at this time, that this structure will be standardized across all instruments. However, parts of the metadata structure, or any (TBD) MODIS produced browse structures, are expected to be standardized.

3.3.2.3.3 Metadata and IMS Access Tools

MODIS will require a facility within the tool kit for MODIS to determine the availability, validity, and content of all ingested Data Products. This currently takes the form of a MODIS specified completion matrix, contained within each Data Product. MODIS expects standardized termination and initiation messages to be specified. These may be free form, to be parsed by the tool kit, or in structure form via headers, etc.

3.3.2.3.4 Telemetry & Command Access Tools

Will a standardized data structure for the command history be applied to all commands or will separate structures for each instrument be available? Instrument commanding requires not only a command history, but a commanded state (instrument state) to be available at each MODIS requested time. A MODIS instrument state would be unique to MODIS and would not be in a standard structure. Perhaps the individual instruments should be responsible for the maintenance of the instrument command history and state based upon command messages received and acknowledged by an auxiliary MODIS processing function.

3.3.2.3.5 Spacecraft Ephemeris & Attitude Data Access Tools

MODIS expects this tool kit function to provide spacecraft position and attitude, interpolated to MODIS specified times. The coordinates of interest are the Earth geocentric and WGS84 ellipsoid coordinate reference frames. Thought should be given to using a standard geoid and providing the intersection of an instrument pointing vector and this reference geoid. DTM is expected to be an iterative, non-invertable function which MODIS will use in appending ground anchor points in both geiod and DTM references, separately. Thought should also be given to providing the instrument pointing vector to DTM intersection as a toolkit function with version and parametric (accuracy) indications as necessary.

The MODIS output Data Products will contain several instances of ground location parameters (anchor points or other form) which may be appended to a Data Product after that product has been made. For example, upgraded spacecraft position and attitude, or ground control point registration will require an updating of ground location indicators which will be appended to the original Data Product, and not replace the original ground location indicators.

3.3.2.3.6 Lunar/Solar/Major Body Position Access Tools

MODIS scientists are interested in Solar Zenith and Azimuth angles of the Sun at (relative to) the intersection of the instrument pointing vector and the Earth as a tool kit function in addition to the mentioned instrument coordinate origins.

3.3.2.3.7 Instrument Calibration Data Access Tools

These calibration parameter files are normal auxiliary data files and do not need special tool kit functions. The structure of these files should be determined by each instrument in conjunction with the instrument characterization teams, not the tool kit.

3.3.2.3.8 Time and Date Access Tools

Time functions and conversions are the subject of 'Holy Wars' and are best met by providing for all 'religions' (flavours). Well written and tested routines for time conversion are a necessity and are rarely available. How are leap seconds, time zones, daylight savings, and formatting to be handled? A generalized parsing function for all data types would also be a nice addition to a tool kit, but is outside the topic of this document.

3.3.2.3.9 Browse Output Tools

The generation of browse products is not currently a MODIS design effort. This is expected to be a DAAC function with browse products being generated on demand only, not in the course of normal processing.

3.3.2.3.10 Quicklook Product Tools

A quicklook indicator in included in the initiation message to each data product generator. In the MODIS case, no special processing is required at the Level-1A and B stages. The MODIS design can handle unordered and duplicate packets in its normal processing.

3.3.2.3.11 Error Output Tools

The MODIS errors that are non catastrophic (processing events) will be logged into the processing log and transmitted to the IMS. These interfaces are planned to be messages that are addressed to the appropriate destinations. Serious problems in the processing will be transmitted to the PGS (scheduler) as termination messages for the PGS to provide resolution, with possible aborting or reprocessing initiated.

3.3.2.3.12 Status Tools

The MODIS design contains a facility for the dynamic query and return of processing status. This will allow the PGS to determine processing progress while the processing is proceeding and to made informed decisions accordingly.

3.3.2.3.13 Data Validation Graphic Output Tools

More detail about the expected tool kit functions that are planned to be provided would be desirable. For example, will an FFT function be available with conversion from the complex domain to a sign magnitude for display purposes? Will a data product generator be suspended while display functions are performed?

3.3.2.3.14 Geographic Standards Tools

MODIS processing requires the ability to determine if an instrument pixel is over land, ice, or water. The land/water value can be obtained from a GIS that contains full polygon attribute determination (is a point inside or outside a closed polygon). Ice determination can not be determined from a GIS and will be determined as one or more Data Products. Also, land/water mixed pixels will pose additional problems.

In addition to terrain elevation, the terrain slope will be required for bidirectional reflectance (BDRF) and water runoff studies. An indication for the tool kit function as to the accuracy and convergence criteria of the returned values will also be required. Mixed pixels may require a deviation value about a one kilometer (for example) spatial area surrounding the requested geographic coordinate.

3.3.3.2 Image Analysis and Manipulation Tools

Image processing functions mentioned in this section can be obtained with the judicious choice of COTS imaging packages. These include the simple tasks of color palettes, image enhancement and equalization, classification, etc. these display functions are analysis tools, not production processing tools. Level 3 and above Data Products will require mapping to user specified mapping projections and parameters. These standardized mapping projections will be required to provide Data Products that are correlatable across various instruments. These projections need to be invertable to allow images from one projection to be transformed into another projection. The data product generators need the ability to select the resampling technique, multiple instrument to single mapped pixel criteria, multiple vector overlay sources, and vector granularity. All of these should apply to both large and small spatial areas of interest. All processing parameters (mapping projection types, origins, resampling techniques, spatial parametrics) must be maintained in the appropriate metadata for each Data Product. Annotated grid and contour lines must also be generated.

3.3.3.3 High Performance Processing Tools

PGS performance enhancements need to be supplied as software equivalents to the SCFs. An example might be to sort a vector array to obtain the largest and smallest numerical values for a feature reduction technique. Digital signal processors (DSP) are available for fast Fourier transforms and other tasks and would need to be emulated locally. Can COTS library functions be implemented in hardware to provide better machine performance?

3.3.4 Data Structure Manipulation Tools

The parsing of a string into its component parts would be a desirable tool kit function. A function that would allow input strings in a flexible form for all possible numeric and character representations and would return a structure with numeric types that the user can select from is one possible implementation. Criteria for parsing would include differing delimiters, comment separators, numeric representations (0xFF or FFh for hexadecimal numbers for example) and precisions (1.0E-5), character field starting and ending positions, and time and date formats (dd/mm/yy or hh:mm:ss for example). The return structure would accommodate various integer and floating point precisions and arrays in the case of time and date.

3.3.5 Memory Management Tools

Computer word ordering and alignment are handled at the individual machine level by the appropriate compilers provided implied data typing is not allowed. For example, implicit none and the form REAL*8 instead of FLOAT are used in FORTRAN and full data declarations of the form 'unsigned long int' instead of 'int' and ANSI function prototyping are used in C. All data sets should be placed into HDF managed file formats. Enforcing these requirements via code checking programs will negate the requirement for this functionality to be a part of the tool kit.

3.3.5.1 Bit, Byte and Word Manipulation Tools

Bit and byte manipulation are handled implicitly with the C constructs and FORTRAN standard libraries provided the above data typing rules are followed.

3.3.5.2 Memory Request Tools

FORTRAN will also need programmable memory allocation routines with the same functionality as the C equivalents. These memory routines may have the ability to hide the location of physical memory granted by the request in order to allow the processing to proceed. Memory allocation that is also a backing store (possible user selectable) may be desirable.

3.3.6 Special Purpose Tools

Mixed C and FORTRAN code in one process can be handled by passing data structures (references) instead of the architecturally dependent, passed by value, approach. If this criteria is not desirable, then a technique needs to be provided in the tool kit such as the gnu f2c or cfortran techniques.

3.3.6.1 Remote Database Access Tools

Will structured SQL be available and sufficiently robust to handle this need?

3.3.6.2

Are you really going to provide all the functions of each DAAC on all the other DAACs? Perhaps the DAAC users don't need the use of auxiliary DAACs when the data access is supposed to be transparent.

3.3.6.3

Standardizing on the free software foundation (FSF) gnu tool set will provide this capability to all users of almost all computers. A donation to FSF would be nice.

Conclusion

MODIS is planning to supply an on-line database containing the information necessary to use all of the functions, utilities, and programs written by the SDST team. This will be automated with key indices for fast access to reusable functions and the avoidance of duplicate code. A similar or combined facility for the PGS would be desirable.